

1	Overview of the Project	1
1.1	Automated Semantic Analysis	2
1.1.1	Oracles	2
1.1.2	Entities and Events	2
1.2	Goals of the Project	3
1.3	Semantic Analysis in TANKA	4
1.3.1	DIPETT Parse Trees	4
1.3.2	Three-Tiered Semantic Analysis	4
	Clause Level Relationships	4
	Case Relationships	5
	Noun Modifier Relationships	5
1.3.3	Background Knowledge	5
	Linguistic Knowledge	5
	Semantic Knowledge	6
	User Knowledge	6
1.4	Evaluation	6
1.4.1	Test Texts	8
1.4.2	Parser Evaluation	8
1.4.3	User Actions	9
1.4.4	User Burden	9
1.5	Applications	10
1.6	Organization of the Dissertation	10
1.6.1	Paper Map	12

1 Overview of the Project

There is no *one-to-one* mapping from syntactic relationships to semantic relationships in English sentences. A sentence with a main clause and a subordinate clause does not always express a causal relationship between two events. The subject of a verb does not always represent the instigator of an act. An adjective does not always indicate a physical property of some entity. Yet there is *a* mapping from syntactic relationships to semantic relationships; some part of the meaning of a sentence is reflected in its surface-syntactic form.

This dissertation is about finding links between syntactic and semantic relationships. In particular it is about defining sets of semantic relationships that can be expressed by syntactically related sentence elements. It is about cataloguing clues that hint at the syntax-to-semantics mapping and using those clues in a program to uncover semantic relationships where they exist in sentences. It is also about recognizing when the clues are insufficient. When this occurs, linguistic facts lurking beneath the surface become necessary; they can be learned from user assistance to make uncovering subsequent semantic relationships easier and more accurate.

1.1 Automated Semantic Analysis

Automated semantic analysis of texts involves such processes as determining word senses and referents, recognizing modification relations and semantic roles, interpreting quantification, and many others. Whatever the processes involved, automated semantic analysis usually attempts to uncover components of the meaning of a fragment of text beyond its surface-syntactic structure. Consider sentences (1) and (2).

- (1) *Ed broke the thumb on Joe's left hand when he swung the hammer off-target.*
- (2) *Ed swung a heavy metal hammer askew and broke Joe's left hand thumb.*

There are many syntactic and lexical differences between (1) and (2). Yet the events and participating entities are similar enough that if both sentences occurred together in a text, one of them might be considered redundant. Semantic analysis should produce structures that reveal some of the underlying similarities of meaning in the sentences.

1.1.1 Oracles

Automated semantic analysis does not necessarily imply full automation or autonomy. Often some outside source of knowledge is required to inform analysis. This oracle may be a large, comprehensive knowledge base, a corpus of analyzed text or a human user—as in this project.

1.1.2 Entities and Events

Since this dissertation deals with semantic relationships marked by syntactic forms, at times I will find it necessary to talk about the semantic *things* that syntactic elements refer to.

Frawley (1993: xiv) identifies certain “basic objects in a model of the world: things, actions, and the relation of things to actions.” *Things* more formally are *entities*: “relatively stable and atemporal discourse, ontological, and conceptual phenomena” (p. 68) and they usually surface as nouns. *Actions* more formally are *events*: “relatively temporal relation[s] in conceptual space” (p. 144) and they usually surface as verbs.¹ Finally, the relation of things to actions is accounted for by *thematic roles* (cases): “grammatically relevant relations between predicates (often events) and arguments (often entities)” (p. 199).

Borrowing liberally from Frawley, then, I will use the term *entity* to refer to the semantic thing represented in syntax as a noun. I will also refer to the semantically complex thing represented in syntax by a noun phrase as an *entity*. Modifiers of a noun in a noun phrase

¹ Frawley's definition of event is general enough to include four more specific kinds: *acts*, *states*, *causes*, and *motion*.

serve to ground it, but the complex noun phrase still refers to a single entity. Using this terminology, both (3) and (4) are entities.

(3) *cat*

(4) *the fat cat on the porch*

Similarly, I will use the term *event* to refer to the semantic thing represented in syntax as a verb. I will also refer to the semantically complex thing represented in syntax by a clause as an *event*. Both (5) and (6) are events.

(5) *snooze*

(6) *the fat cat on the porch snoozes all day long*

1.2 Goals of the Project

The semantic relationships in the sentences of a text form a model of that text. With the present state of the art, fully automatic construction of such a model is not feasible without a knowledge base. A fully manual construction of the model is feasible, but onerous. My thesis is that it is possible to acquire the model interactively without demanding of a user as much as a large knowledge engineering effort. This claim can be tested through the following specific project goals. The extent to which the goals have been met will be investigated in chapter 6.

- 1 *To construct usable, portable sets of semantic relationships.* The sets should take into account similar sets proposed by others and should be general enough for any technical text and for other kinds of relationship-based analysis of texts. Construction should be informed by empirical considerations, such as marker data and text coverage.
- 2 *To design and implement a semi-automatic system for recognizing semantic relationships in text.* The system should make use of whatever grammatical knowledge is available, but should avoid open-ended, domain-specific semantic knowledge. It should also offer informed suggestions and learn from user input.
- 3 *To evaluate the ability of the system to learn.* As more text is processed, the number of semantic relationships recognized automatically should increase, reducing reliance on the user.
- 4 *To evaluate the system's coverage of relationships in a text.* Parse tree input is not always perfect. The system should be able to recognize some semantic relationships even from imperfect parses.

- 5 *To evaluate the burden the system places on the user.* Recognizing semantic relationships is not always easy, even for people. The system should be able to analyze reasonable amounts of text in a reasonable amount of time without overtaxing the user.

1.3 Semantic Analysis in TANKA

TANKA (Text ANALYSIS for Knowledge Acquisition) is a project whose goal is a system that produces a model of the knowledge contained in a technical text. Syntactic analysis in TANKA is performed by DIPETT; semantic analysis is performed by HAIKU. The design, implementation and evaluation of HAIKU are the vehicle for accomplishing the goals listed in the previous section.

1.3.1 DIPETT Parse Trees

DIPETT (Domain Independent Parser of English Technical Texts) is a broad-coverage Definite Clause Grammar parser (Delisle 1994, Delisle & Szpakowicz 1995) whose rules are based primarily on Quirk *et al.* (1985). DIPETT automatically produces a single initial parse. This first good parse tree is a detailed representation of the constituent structure of a sentence. In particular, the parse tree contains the noun phrases, clauses and syntactic connections between clauses that make up the input for HAIKU semantic analysis.

1.3.2 Three-Tiered Semantic Analysis

Semantic analysis in TANKA is the semi-automatic recognition of semantic relationships among entities and events represented in sentences. HAIKU recognizes three kinds of semantic relationships: clause level relationships, cases and noun modifier relationships. This division of semantic relationships is motivated by their syntactic manifestation at different levels. It is possible, however, for the same semantic relationship to be realized at any of the levels, suggesting that a unified view of semantic relationships is appropriate. For this project the relationships that apply at each level remain separate, and I investigate the possibility of unifying them in section 5.4.

At each level HAIKU attempts to find the best relationships using syntactic evidence and previous analyses. The system suggests these relationships to a cooperating user for approval. As more sentences are analyzed, HAIKU learns to make better suggestions based on the user's input.

Clause Level Relationships

Clause level relationships (CLRs) are semantic relationships between events represented syntactically by syndetically connected finite clauses. The CLR analyzer recognizes these relationships in sentences by considering the lexical items that mark them (coordinators, correlatives, subordinators) and the syntactic features of the connected clauses.

Case Relationships

Cases are semantic relationships that express the roles of the participants and the circumstances of an event. The case analyzer recognizes cases in finite clauses by considering the syntactic or lexical items that indicate verb arguments (subject, objects, prepositional phrases, adverbials).

Noun Modifier Relationships

Noun modifier relationships (NMRs) are semantic relationships that define entities. They are expressed within noun phrases as modification. The NMR analyzer recognizes relationships expressed by adjectives, premodifying nouns and postmodifying prepositional phrases and appositives.

1.3.3 Background Knowledge

HAIKU can be characterized as knowledge-scant. It is designed to avoid excessive reliance on precoded, domain-dependent, semantic knowledge. In practice certain kinds of knowledge encoding are forbidden in TANKA and certain kinds are allowed. The distinction is not entirely arbitrary. In this section, I will describe what knowledge *is* allowed in TANKA, and how it differs from the knowledge that is not.

Linguistic Knowledge

Both DIPETT and HAIKU use all kinds of linguistic knowledge. DIPETT's grammar is linguistic knowledge. The fact that part of the meaning of an utterance is reflected explicitly in its surface structure (*e.g.*, that an entity is represented syntactically as a noun and an event as a verb) is linguistic knowledge. That modal auxiliaries express uncertainty, that prepositional and adverbial phrases indicate circumstances of an event, that certain kinds of noun compounds are hyponyms of their heads are all bits of linguistic knowledge, and are all allowed by TANKA.

These examples are knowledge of language in general. Such knowledge is closely tied to the surface syntax and does not vary from domain to domain or from text to text. It is independent in the sense that each linguistic fact can be encoded to assist the semantic analysis task without requiring that further knowledge (such as lexical semantics) be added for the fact to be useful.² Furthermore, the linguistic knowledge in TANKA is relatively uncontroversial. There may be disagreement on the specific semantic relationships possible between adjectives and the nouns they modify, but there is relatively less disagreement on the fact that such relationships exist.

² An example of linguistic/semantic knowledge that does not have this property is the knowledge that nouns denoting animate objects are more likely instigators of acts. That knowledge would require that nouns in a lexicon be tagged as \pm animate.

Semantic Knowledge

The sets of semantic relationships recognized by HAIKU are semantic knowledge (sections 2.3, 3.4 and 4.4). The dictionaries mapping lexical markers to subsets of relationships are semantic knowledge (sections 2.4, 3.3.1 and 4.5). The preference rules for clause level relationships and the paraphrases for noun modifier relationships are semantic knowledge (sections 2.5.3 and 4.4.1).

The semantic knowledge allowed in TANKA must be closed. Mapping markers to semantic relationships is allowed since the sets of markers and relationships are small, fixed and domain-independent: they do not grow or change from domain to domain. Tagging nouns with semantic class information is not allowed in TANKA, since the nouns are an open category and their senses may change from domain to domain. The number and nature of preference rules for CLR depend on the CLR, not on the semantics of the main verbs in clauses. For example, it might be tempting to add a rule stating that a clause with the main verb *prevent* is evidence for the Prevention CLR (section 2.3.1), but such rules would be potentially open-ended. Case assignment would be relatively simple if the case patterns associated with particular verbs were known in advance. Again, that knowledge is open-ended, and not allowed in TANKA.

The techniques used by HAIKU to recognize semantic relationships should be independent of the allowed semantic knowledge. The noun modifier relationship analyzer can work with any set of noun modifier relationships, and even allows run-time addition of new NMRs (section 4.6.5). CLR analysis uses the preference rules for pairs of CLR if such rules exist. If not, the CLR are given equal weight in CLR competitions (section 2.5.4), meaning that HAIKU could assign CLR even with no prior knowledge of CLR semantics. In the absence of marker dictionaries, HAIKU would be more dependent on user input, but could still improve over time by learning from user assignments.

User Knowledge

In the absence of other kinds of semantic knowledge, HAIKU relies on the user to supply information. It is therefore a priority in HAIKU to minimize the burden placed on the user in interactions with the system. Evaluating the decrease in user activity over the course of analyzing a text measures HAIKU's success in curbing user burden.

1.4 Evaluation

A significant part of this dissertation is devoted to the evaluation of HAIKU's performance on various texts and of the knowledge it acquires. The evaluation experiments were necessarily specific to TANKA, but I have also paid attention to some more general evaluation principles.

There has been increased interest in the evaluation of natural language processing (NLP) tools over the last few years. The message understanding conferences (MUCs) are a direct reaction to a lack of standardization in the evaluation of such tools. The MUC competitions compare the performance of different systems on some uniform, predetermined text processing task. The motivation for and background of the MUC competitions is described in Grishman & Sundheim (1996).

Although these competitions have successfully emphasized the importance of evaluation for the NLP community, they have also revealed certain limitations. In particular, researchers have noted that the predefined tasks in evaluation competitions result in applications that are designed to score well, but are not portable (see MUC-6 1996). Furthermore, little attention has been paid to the evaluation of interactive systems and the role of users (Hirschman & Thompson 1996).

To address these issues, Sparck Jones (1994) and Sparck Jones & Galliers (1996) offer more general strategies for evaluating generic NLP systems: “there is far too much variety in the situations and subjects of evaluation to come up with a definite [evaluation] scenario” (Sparck Jones & Galliers 1996: 193).

Hirschman & Thompson (1996) distinguish two kinds of evaluations—*diagnostic evaluations* and *performance evaluations*—that make different assumptions about the distribution of test data. A diagnostic evaluation tests a system on all of the linguistic phenomena that it is designed to handle. Lehmann *et al.* (1996) describe the construction of an exhaustive test suite for diagnostic evaluations of natural language processing systems. The distribution of linguistic phenomena in the test suite of a diagnostic evaluation is almost certainly not the same as the distribution of phenomena in complete texts, which form the test suite for a performance evaluation. For performance evaluations it is important to identify *criteria* (what is being evaluated), *measures* (what properties of performance reflect the criteria) and *methods* (how the measures are analyzed to arrive at an evaluation of the criteria).

The components of HAIKU will be evaluated on four general criteria:

- 1 the number of semantic relationships recognized correctly by the system;
- 2 the increase in proportion of correct analyses over the course of a session (HAIKU’s ability to learn);
- 3 the system’s coverage of the test texts (number of semantic relationships recognized relative to the number actually in the text);
- 4 the burden that interaction places on the user.

The first two criteria are objective and can be measured directly. Coverage is objective, but cannot always be measured directly, since the number of semantic relationships in an entire text is not always easy to determine. I will use sampling when direct measurement is not feasible (see sections 3.6.1 and 4.7.2). The fourth criterion is mostly subjective, but

its evaluation will take into account quantitative measurements of user participation (see sections 1.4.3 and 1.4.4).

1.4.1 Test Texts

TANKA targets unedited English technical text for semi-automatic analysis. Technical texts usually lack humour, intentional ambiguity, and other devices that would make analysis more difficult. Copeck *et al.* (1997) describe a study to investigate the character of texts typically considered technical.

DIPETT and HAIKU have been tested on a variety of technical texts. I refer to four of these experiments repeatedly throughout the dissertation.

The *clouds* experiment was a performance evaluation summarized in Barker & Delisle (1996). The text for the experiment was the *Junior Science Book of Rain, Hail, Sleet & Snow* (Larrick 1961). The *clouds* text has fairly simple syntax and was chosen to ensure a high number of correct parses. The belief was that the higher parse success rate would allow the system to recognize more semantic relationships, since HAIKU would have access to more correct parse trees. Unfortunately, the simpler text uses very general terminology and less complex modification, resulting in fewer semantic relationships. The *clouds* experiment was meant to evaluate DIPETT, CLR analysis and case analysis as well as user burden. Sylvain Delisle and I were the two users in the experiment.

Results of the *small engines* experiment appear in Barker *et al.* (1998). *The Mechanics of Small Engines* (Atkinson 1990) has more complex syntax, resulting in fewer correct parse trees available for HAIKU. On the other hand, the text contains more semantic relationships per sentence. This experiment was a performance evaluation of DIPETT, CLR analysis, case analysis, NMR analysis and user burden. Sylvain Delisle and I were again the users in the experiment.

The *building code* experiment used sentences from the Ontario Building Code (Ontario Ministry of Housing 1991) in a diagnostic evaluation of the CLR analyzer. The *building code* text contains thousands of complex sentences with a variety of conjunctions connecting clauses and a variety of syntactic verb phrase features.

The *sparc* experiment was a performance evaluation of components of the NMR analyzer on noun phrases from the *SPARCstation 1 Installation Guide* (Chan 1989).

1.4.2 Parser Evaluation

For the *clouds* and *small engines* experiments, the quality of DIPETT's output had a major effect on semantic analysis.

The *clouds* experiment applied DIPETT and HAIKU to 512 sentences. DIPETT produced a single complete parse tree covering all tokens (words and punctuation) for 412 sentences (80%). A complete parse does not necessarily mean a correct parse. 47% of the parses of *clouds* sentences were perfect. Another 19% were good enough not to affect semantic

analysis. The other 34% of the parse trees had errors that were serious enough to prevent the recognition of one or more semantic relationships by HAIKU.

In the *small engines* experiment, 55% of the 557 sentences received a complete parse. Due to the much more complex syntax, only 31% of the parses were perfect and another 9% were good enough not to affect semantic analysis. That means that for 60% of the sentences in the *small engines* text, parse errors were serious enough to prevent HAIKU from recognizing all the semantic relationships.

The individual evaluations of CLR analysis (section 2.6), case analysis (section 3.6) and NMR analysis (section 4.7) investigate the extent to which HAIKU was affected by parse errors.

1.4.3 User Actions

The evaluation of HAIKU's success at recognizing semantic relationships will be judged by the type of action required by the user. Every time HAIKU assigns a semantic relationship to an input, it requires approval from the user, so even correct analyses involve user interaction.

The first type of user action is *accept*: HAIKU presents the user with one single semantic relationship suggestion and the suggestion is correct. *Accept* actions are considered correct semantic relationship assignments by the system.

The second type of action is *choose*: HAIKU suggests more than one possible semantic relationship and the correct relationship is among them. The number of suggestions is limited to the number of marker→relationship mappings for a given marker, which is at most roughly half the total number of semantic relationships. In practise the number of suggestions is usually no more than three or four (see for example sections 3.6.1 and 4.7.2).

The third action is *supply*: either HAIKU is unable to offer any suggestions, or the correct relationship is not among HAIKU's suggestions; the user must supply the relationship.

In certain instances it will be convenient to compare the number of *supply* actions to the sum of the other two types. In those comparisons I will use the term *system assignments* to refer to semantic relationships assigned as a result of a *choose* or *accept* action and *user assignments* to refer to relationships assigned as a result of a *supply* action.

1.4.4 User Burden

For each user interaction in the *clouds* and *small engines* experiments, the degree of difficulty of interaction (referred to as *onus*) was recorded as an integer from 0 to 3. 0 means that the interaction is trivial. 1 is assigned to an interaction that requires a few moments of reflection. 2 rates an interaction as requiring serious thought, but eventually a semantic relationship is assigned. 3 means that even after much contemplation no relationship is deemed appropriate for the given input.

User burden is also reflected in the amount of time required to oversee the analysis of a text. The average number of tokens (words or punctuation) in the *small engines* sentences was 15.4. The average number of CLRs per sentence was 0.04, while the average number of case patterns was 1.05 and the average number of NMRs was 1.59. On average, we spent 1 minute, 49 seconds on each sentence. By coincidence, the average user time for the *clouds* experiment was also 1 minute, 49 seconds for an average of 0.10 CLRs and 0.86 case patterns per sentence (NMRs were not evaluated in the *clouds* experiment).

1.5 Applications

Information extraction from text often involves the identification of essential participants and circumstances in acts. The participants and circumstances can be represented directly as cases. More specific knowledge of the participants involves recovery of the relationships within noun phrases (NMRs).

Template filling is a kind of information extraction. A template has a fixed set of slots to be filled with specific information from the text. For events, these slots are often analogous to cases (with names such as Agent, Instrument, Location, etc.) and sometimes also relate events to other events (comparable to recognizing CLRs). For entities the slots often correspond to hypernyms, subcomponents, purposes and properties—knowledge of the kind that is captured by NMR analysis.

It is often claimed that case is a universal phenomenon across natural languages (Fillmore 1968). Research has identified strikingly similar lists of cases for many human languages (see Campe 1994 for a multilingual bibliography of case). The semantic roles in a source language clause could be used as part of an *interlingua in machine translation*.

Question answering systems are often faced with questions about properties of entities and the participants and circumstances of events.

The construction of *semantic lexicons* has become a popular area of research in natural language engineering. Noun entries in these lexicons often identify properties of entities, hypernyms, subcomponents, as well as the events in which the entity expressed by the noun participates.

1.6 Organization of the Dissertation

The five goals of section 1.2 apply to all three levels of processing in HAIKU. This dissertation is divided into three main chapters corresponding to HAIKU's clause level relationship analysis, case analysis and noun modifier relationship analysis. Each chapter describes work done towards satisfying the project goals: building a set of semantic

relationships; implementing a semi-automatic system that uses linguistic clues to recognize semantic relationships in sentences; evaluating the system on its ability to learn, the coverage of its relationships and the burden it places on the user.

Clause level relationship analysis, since it deals with the largest (and most intricate) parse tree fragments, has access to more linguistic evidence for recognizing relationships. The connection between syntax and semantics is stressed more in chapter 2 than in the other chapters, simply because there is more syntax at that level than the others.

Case analysis is not part of the contribution of this project, since the techniques have already been described in Delisle (1994) and Delisle *et al.* (1996). What *is* part of this project is the process of constructing the set of cases and the evaluation of their coverage. HAIKU's cases have received more attention in evaluation than either CLR's or NMR's. Chapter 3 is primarily about defining and evaluating a set of semantic relationships.

There is little surface-linguistic evidence to guide the assignment of noun modifier relationships. The NMRs themselves have not been tested for coverage to the extent that the cases have been. What is unique about the NMR analyzer is its use of partial matching on a growing base of previous instances, and the attention paid to elements of the user interface. Chapter 4 has an emphasis on learning and user interaction that does not occupy as prominent a place in the other chapters.

Chapter 5 explores reasonable extensions to the project along with bolder departures. Chapter 6 summarizes the project and evidence that its goals have been met. Three appendices contain samples of the clause level relationship marker dictionary (see section 2.4), the case marker dictionary (section 3.3.1) and the noun modifier relationship marker dictionary (4.5). Literature review has been parcelled out to sections 2.1, 3.1 and 4.1. There is an abundance of interesting work related indirectly to each of the three main areas of the project, but less that is directly applicable.

1.6.1 Paper Map

Parts of the dissertation have already been the subject of technical reports and papers:

<i>topic</i>	<i>dissertation sections</i>	<i>previously described in</i>
CLR Analysis	2.1-2.4, 2.5.1-2.5.4, 2.6.1, 2.6.2	Barker (1994), Barker & Delisle (1996), <u>Barker & Szpakowicz (1995)</u> , <u>Barker et al. (1998)</u>
Cases	3.1-3.4, 3.6.2	Barker (1996), Barker & Delisle (1996), <u>Barker et al. (1997)</u>
Case Analysis	3.5, 3.6.1	Barker & Delisle (1996), <u>Barker et al. (1998)</u> , <u>Delisle et al. (1993)</u> , <u>Delisle et al. (1996)</u> ,
NMR Analysis	4.1, 4.2, 4.4-4.6, 4.7.2	Barker (1997), <u>Barker & Szpakowicz (1998)</u> , <u>Barker et al. (1998)</u>
Bracketing	4.3, 4.7.1	Barker (1997), <u>Barker (1998)</u> , <u>Barker et al. (1998)</u>
Evaluation	2.6.1, 2.6.2, 3.6, 4.7	Barker & Delisle (1996), <u>Barker et al. (1997)</u> , <u>Barker et al. (1998)</u>